Running Example Liskov Substitution Principle (LSP)

Flavianus Putratama - 21120122140105

Abstract

Liskov Substitution Principle (LSP) adalah salah satu dari lima prinsip SOLID dalam pemrograman berorientasi objek. Prinsip ini menyatakan bahwa objek dari kelas turunan harus dapat menggantikan objek dari kelas induknya tanpa memengaruhi fungsionalitas program. Dengan kata lain, jika kelas B adalah subclass dari kelas A, maka objek dari kelas B harus dapat digunakan di mana saja objek dari kelas A digunakan, tanpa mengubah perilaku yang diharapkan. Hal ini memastikan bahwa subclass memperluas, bukan mengubah, perilaku dari superclass.

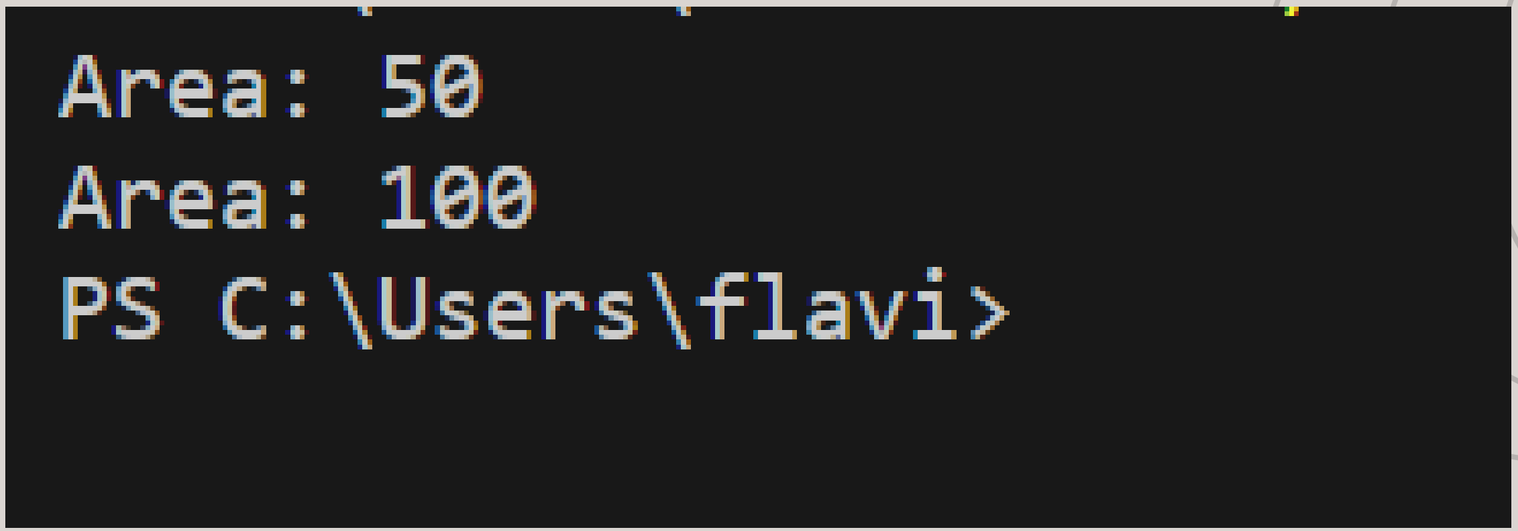
Demo Problem

Demo Problem Liskov Substitution Principle (LSP) .py

F: > Downloads > Demo Problem Liskov Substitution Principle (LSP) .py > Rectangle > get_area

```python
# Superclass Rectangle
class Rectangle:
    def __init__(self, width, height):
        self._width = width
        self._height = height

    def set_width(self, width):
        self._width = width

    def set_height(self, height):
        self._height = height

    def get_area(self):
        return self._width * self._height

# Subclass Square yang melanggar LSP
class Square(Rectangle):
    def __init__(self, side):
        super().__init__(side, side)

    # Override set_width agar sisi selalu sama
    def set_width(self, width):
        self._width = self._height = width

    # Override set_height agar sisi selalu sama
    def set_height(self, height):
        self._width = self._height = height

# Fungsi untuk menghitung area dari objek Rectangle
def calculate_area(rectangle: Rectangle):
    rectangle.set_width(5)
    rectangle.set_height(10)
    print(f"Area: {rectangle.get_area()}")

# Testing
rect = Rectangle(2, 3)
calculate_area(rect)  # Output: Area: 50 (sesuai dengan panjang dan lebar)

square = Square(5)
calculate_area(square)  # Output: Area: 100 (tidak sesuai dengan yang diharapkan)

```

Demo Output Problem



Kelas **Square** melanggar LSP karena ketika kita mengatur lebar **(set_width)** atau tinggi **(set_height)**, kedua sisi harus selalu sama. Ini menyebabkan perilaku yang tidak diharapkan saat menghitung area di fungsi **calculate_area()**. Misalnya, setelah memanggil **set_width(5)** dan **set_height(10)**, kita mengharapkan area menjadi **50**, tetapi karena **Square** memperlakukan lebar dan tinggi sama, area sebenarnya adalah **100**.

Demo Solver

Demo Problem Liskov Substitution Principle (LSP) .py | Demo Solver Liskov Substitution Principle (LSP) .py

F: > Downloads > Demo Solver Liskov Substitution Principle (LSP) .py > ...

```python
# Superclass Shape (bentuk umum)
class Shape:
    def get_area(self):
        raise NotImplementedError("This method should be overridden")

# Subclass Rectangle
class Rectangle(Shape):
    def __init__(self, width, height):
        self._width = width
        self._height = height

    def set_width(self, width):
        self._width = width

    def set_height(self, height):
        self._height = height

    def get_area(self):
        return self._width * self._height

# Subclass Square
class Square(Shape):
    def __init__(self, side):
        self._side = side

    def set_side(self, side):
        self._side = side

    def get_area(self):
        return self._side * self._side

# Fungsi untuk menghitung area dari objek Shape
def calculate_area(shape: Shape):
    print(f"Area: {shape.get_area()}")

# Testing
rect = Rectangle(5, 10)
calculate_area(rect)  # Output: Area: 50

square = Square(5)
calculate_area(square)  # Output: Area: 25
```

Demo Output Solver

```
Area: 50
Area: 25
PS C:\Users\flavi>
```

Dengan kode solver ini, **Rectangle** dan **Square** adalah subclass dari **Shape**, tanpa hubungan inheritance di antara mereka. Mereka masing-masing mengimplementasikan perilaku area yang sesuai. Fungsi **calculate_area()** sekarang dapat bekerja untuk **Rectangle** dan **Square** secara terpisah, sesuai dengan sifat masing-masing bentuk. Dengan ini, kita mematuhi LSP karena **Square** dan **Rectangle** tidak saling menggantikan dengan perilaku yang tidak sesuai.

Thank You